What is HCI?

Definition: HCI is about the design and use of computer technology Focus: The interaction between humans and computers through interfaces.

What is Design? Making things with a specific goal

What is an Interface? The place at which independent and often unrelated systems meet and act on or communicate with each other

Design Process Steps: Talk to people, investigate problems. Sketching and Storyboarding. Low-fidelity Prototyping. Digital Mockup. Presentation & Communication.

Core Design Principles: Usability, Learnability, Efficiency, Safety, Accessibility, Aesthetics: Visual appeal and minimalism. Ergonomics: Physical comfort and fit Expressivity: Flexibility for different uses Malleability: User control and customization

Design Trade-offs and Context: Principle Emphasis Depends on Users, Principle Emphasis Depends on Tasks.

User-Centered Design Process: has four main phases: Frame the Problem - User research, competitive analysis. Explore Solution Space - Brainstorming, ideation, sketching. Find a Good Solution - Scenarios, storyboards, personas, design rationale. Refine the Solution - Wireframes, prototypes, evaluations

Iterative Design vs. Waterfall: Problems with Waterfall Model: UI design is risky - high chance of getting it wrong first time, Users not involved until end - validation only at acceptance testing, UI flaws cause requirement changes - expensive to fix late in process, Linear progression - Requirements → Design → Code → Integration → Acceptance → Release

Benefits of Iterative Design: Continuous cycle: Design → Implement → Evaluate → repeat Risk management, Spiral model, Early prototyping catches problems when they're cheap to fix

Pros of Iterative Design: Early iterations use cheap prototypes, More iterations = better UI, Only mature iterations seen by public

Design Diamond: Two Key Phases: Elaboration (Divergent thinking) - Generate MORE ideas Reduction (Convergent thinking) - Refine DOWN to best ideas

Critical Points: Intentional transitions between generate and select phases, don't select too early, Both good and bad ideas are useful, Critique helps

Prototyping Progression: 30 sketches → 5 paper prototypes → 2 digital prototypes. Diamond gets smaller over time. Parallel design early - multiple alternatives simultaneously Serial design later - one refined design at a time

Types of Prototyping Early Prototyping Examples: Sketches - Quick, disposable, explore many alternatives Paper prototypes - Interactive, simulate user flows Wireframes - Layout structure without visual design Digital mockups - Higher fidelity, closer to final product

Parallel vs. Serial Design: Parallel design: Multiple prototypes simultaneously Serial design: One design at a time

Ideation and Sketching; Purpose of Sketching: Think through ideas early in design process, Explore alternatives quickly and cheaply, Convey ideas to others for feedback, Easy to make, Quantity over quality.

Ideation Techniques: Challenge assumptions, Explore design dimensions, Push boundaries, Design space exploration.

Design Phase: Observation - understand user problems, Idea generation, brainstorming, Sketching.

Implement Phase: Paper prototypes, Video prototypes, Wireframes, Digital prototypes.

Evaluate Phase: Critique, Wizard-of-Oz testing (simulate functionality manually), Heuristic evaluation (expert review against principles), User studies.

Design Diamond Structure: Ideation (generate many ideas) → Critique (select best ideas) → Prototype. Process: Start broad, then narrow down through feedback.

What is Design Fixation? Definition: Getting stuck on limited design approaches due to examples or past experiences Causes: Examples shown too early in process, past experiences and mental associations, cultural/personal biases.

How to Overcome Design Fixation 1. Diverse Examples, 2. Quantity over Quality, 3. Mapping the Design Space, 4. Parallel Prototyping: Create multiple different prototypes simultaneously. Benefits: Better final designs, More diverse solutions, Higher user engagement metrics, Less emotional attachment to single idea.

Socratic Method: Identify design aspect and ask "Why?", Forces presenter to explain/develop rationale, Good when unsure what else to say.

Design Principles for Learnable UIs

1. Affordances Definition: Visual cues that show how something can be operated

2. Feedback Principle: Actions should have immediate visible effects. Types: Low-level, High-level.

3. Recognition vs. Recall Recognition: Remembering with visual cues, Recall: Remembering without help

4. Multiple Interaction Styles Provide different ways to accomplish same task

5. Consistency Internal Consistency: Within your application, External Consistency: With other applications, Rule: Similar things should look/act similarly, different things should look different

6. Metaphors Purpose: Bring outside world knowledge into interface, Benefits: Highly learnable when appropriate, hooks into existing mental models Risks: Can be constraining, break down at some point, may become outdated.

7. Mapping Principle: Physical arrangement of controls should match arrangement of function

8. Visibility and State Exposure; **Visible Selection State**: Highlight selected objects, Provides feedback and shows current state, Don't leave selections invisible/implicit.

Visible Navigation State: Breadcrumbs: Show current location in hierarchy, Pagination: Show current page and available pages, Tabs: Highlight active tab

Self-Disclosure: Make command language visible through GUI

Visible Modes: Modes: Same action, different results, Use spring-loaded modes when possible, Provide strong visual cues for current mode.

How Learning Breaks Down; **Three Models in UI Design**: System Model (Implementation Model): How system actually works, Interface Model (Manifest Model): Model interface presents to user, User Mental Model (Conceptual Model): How user thinks system works.

Model Mismatches: Interface should hide system complexity, User mental model may be wrong, Designer's job: Make interface model clear and match user expectations.

Norman's Execution/Evaluation Cycle Execution Side (Gulf of Execution): Goals → Form Intention → Develop Action Plan → Execute Actions, Evaluation Side (Gulf of Evaluation): 4. System Change → Observe State → Interpret State → Evaluate Goals

Common UI Problems: Poor Affordances, Feedback Issues, Visibility Problems, Model Mismatches.

Slips and Lapses; Definition: Failure in executing a skill user already learned. **Slip**: Failure due to execution/control. Example: Missing button on click, Ctrl-V instead of Ctrl-C.

Lapse: Failure due to memory. Example: Forgetting to add attachment to email.

Mistakes; Definition: Error made in planning or rule execution. Different from slips/lapses - involves wrong planning, not wrong execution.

Types of Slips; **Capture Slip**: Person starts one action sequence but veers into another (more familiar) sequence. Example: Walking to school instead of intended destination.

Description Slip: Two actions are very similar, user substitutes wrong one. Example: Reaching for milk but grabbing orange juice.

Mode Error Modes = states where same action has different meanings. User forgets which mode they're in. Example: Typing with Caps Lock enabled when wanting lowercase.

Preventing Errors; **Remember Consistency**: Similar things should look/act similarly, Different things should look different, Keep dangerous commands away from common ones.

Modes: Generally Eliminate. If must use modes: Increase visibility of modes, Use spring-loaded/temporary modes.

Confirmation Dialogs; Problems with confirmation dialogs: Reduces efficiency (requires 2 actions instead of 1), Expert users learn to habitually press OK without reading, Better solution: Reversibility (undo).

When to use confirmation dialogs: Only for rare, catastrophic events, Make it look very different from everything else, Draw attention: no generic OK button, force user to think.

Chunking; Chunk: Unit of memory/perception, Working memory limited to 4 chunks $- >$ 10 second. Design implication: Group information to improve efficiency.

FITTS' LAW Formula: $T = a + b \log(D/S + 1)$, T: Time to acquire target, D: Distance to target, S: Size of target, a: Reaction time, b: Throughput (bandwidth human→computer).

Index of Difficulty (ID): $ID = \log(D/S + 1)$ As D increases → difficulty increases As S increases → difficulty decreases Units don’t matter (it’s a ratio)

Key Implications; Screen Edges & Corners: Targets at screen edge are easy to hit, Save edge positions for frequent actions, Unclickable margins are bad design, Target Size & Distance: Make frequently-used targets BIG, Put targets used together NEAR each other, Use screen corners and edges for important functions, Steering Tasks: Steering is much harder than pointing, Constrains error size while moving toward target, Cascading submenus are hard to use. Fitts’ Law Takeaways: Make frequently-used targets big, Put related targets near each other, Use screen corners and edges, Avoid steering tasks.

Key Principles of User Research:

General Guidelines: Make the environment comfortable for subjects, Look for implicit insights (what people do, how they behave) behind explicit ones (what they say), Respect subjects, accommodate them, Keep an open mind - look without presupposing what you’re looking for, Trust that your ability to define the problem will emerge during the process.

Critical Design Lesson: Disability Dongle: Definition: A well-intended, elegant, yet useless solution to a problem we never knew we had, Key Quote: ”To do this work, the first rule you have to teach yourself is that you are not your user”, Problem: Designers creating solutions without consulting the actual user community they claim to help.

Research Methods Overview:
Observational - watching what people do, Self-Report - asking what people think/say.

Observational Method;

Participant Observation: Goal: Viewing users and their behavior in context When: Want to see users in their element and learn about their experience How: ”Deep hanging out” - spend time near subjects Active participation in their activities Experience membership in the context/culture Form connections and empathy

Fly-on-the-Wall Observation: Goal: Deep understanding of how people behave in specific location When: Want to study people unobtrusively (avoid bias/Hawthorne effect) How: Go to location and observe without interacting - be invisible

Behavioral Mapping: Goal: Uncover discrepancies between how participants use space vs. intended use, When: Want to study specific space/environment to plan improvements, How: Unobtrusive observation ”at a distance”, Start with site plan/map, List behaviors to record, Note when behaviors happen on the map.

Self-report Methods:

Interviews: Goal: Collect firsthand personal accounts of experiences, opinions, attitudes, perceptions. When: Want to maximize info per time spent, likely to have follow-ups, context less important. How: Semi-structured script with flexibility to deviate. Limitations: People bad at remembering specific details, People remember almost nothing about routine events, People bad at estimating frequency.

Directed Storytelling: Goal: Collect rich stories of people’s experiences. When: Want stories when time/factors prevent direct observation or longer research. Good when: You wouldn’t be able to come up with interview questions (need more knowledge). How: Focus on stories.

Focus Groups: Goal: Gain insights into themes, patterns, trends that emerge in group settings. When: Want to learn opinions/feelings/attitudes from group; interested in how people build off each other’s thoughts. How: Interview script with small group, be flexible to let group carry conversation.

Surveys: Goal: Collect large amounts of self-reported information. When: Want many perspectives rapidly and/or quantify insights; have good understanding of questions to ask. Benefits: Can be remote, statistical testing possible. Limitations: Less flexible than interviews, requires more knowledge about population, no follow-ups.

Diary Studies: Goal: Capture specific details of real interactions close to when they happened. When: Want to learn about specific activities over extended time period. Benefits: Capture specific context (photos, screenshots), Better understand frequency of use, Good when can’t observe users throughout time period. Note: Less frequent activities can take months to document

Graffiti Wall: Goal: Collect participants’ responses/thoughts on environment/system directly in context. When: Want to collect info where interviews/observation might be unsuitable. How: Provide open canvas for participants to freely write or draw.

Mixed Methods:

Contextual Inquiry: Combines: Both observation and self-report. Core Premise: ”Go where the customer works, observe as they work, talk about the work”. Goal: Study actual behavior in real contexts of use; see how context impacts interactions. When: Not worried about biases from observation/discussion; interested in specific tasks. Process: Define tasks up front, Conduct session in normal place/time, Let participant complete task with few interruptions, Have them think aloud, Ask questions afterward. Back-and-forth structure: Participant does task in appropriate context, Participant explains what they’re doing, Researcher offers interpretation, Participant agrees or corrects.

Personal Inventories Mix: Observation and self-report. Goal: Understand relationship between product and users from participant’s POV. When: Want to explore relationship between user and specific artifact/product. How: Prepare questions about things they own, Ask participants to walk through inventories, Ask specific questions about relationships/emotions.

Interview Best Practice: Interview Structure (Timeline): Intro: Introduce yourself and study, set expectations (”no right/wrong answers”). Kick-off: Simple opening question. Build Rapport: Easy, personal questions to get comfortable. Grand Tour: Main substantive questions, stories, experiences. Reflection: Hypothetical or design-oriented questions. Wrap-Up: Thank participant, next steps. Interview Tips: Open-ended, semi-structured, Guide but don’t dominate, Allow silence, Avoid bias, Don’t interrupt.

Design Thinking (IDEO Method): Definition: 5-step design process: Empathize → Define → Ideate → Prototype → Test. Philosophy: ”We’re experts on the process of how you design stuff” - Dave Kelley. Non-linear process: Can iterate between steps.

Critiques of Design Thinking: Not foolproof: Can’t solve every problem with design. Wicked problems: Complex issues with incomplete/contradictory requirements. Avoid technological solutionism: Not every problem needs a tech solution. Shallow understanding risk: Don’t helicopter in with quick fixes. Need domain expertise: Include diverse backgrounds and local knowledge.

Design Empathy: Definition: Purpose: ”Deep emotional understanding of people’s needs to inspire innovation”. Method: Experience users’ situations firsthand.

Critiques of Design Empathy: Not always appropriate: Don’t blindfold sighted people to test for blind users. Why problematic: Blind people have developed different skills over lifetime. Limitations: Any empathy attempt is incomplete. Best use: For temporary experiences or similar contexts. NOT a replacement: Still need actual user research and user feedback.

Final User Research Tips:
1. Cast Aside Biases, Listen and Observe: Let subjects tell their own story, Listen for emotional triggers (concern, frustration), 2. Note Contradictions Between Words and Actions: Key insight: ”Opportunities for innovation lie within the disconnect between action and words”, Watch what people actually do vs. what they say they do. 3. Listen to People’s Personal Stories: Let them relate successes and failures, Show what people find normal, acceptable, true, Reveal moral codes, sources of pride, shame. 4. Watch for ”Work Arounds”: People adapt to product/situation shortcomings, We create clumsy or clever solutions unconsciously, Critical: You must take note of these adaptations. 5. Distinguish Between Needs and Solutions: Needs: Open up possibilities, Solutions: Constrain possibilities, Starting with solutions may cause you to miss revolutionary alternatives. 6. Look Beyond the Obvious: Research may seem routine and familiar, Stay alert to avoid boredom and frustration, Key insight: ”Epiphanies and insights emerge from the nuances”.

Task Analysis Overview: Definition: Purpose: Understand how people accomplish specific tasks Goals: Identify tasks your solution must support, Find simplest, most effective task completion methods, Use with other user research methods as a lens.

Knowledge Types Gathered: Declarative Knowledge: Objects and relationships. Procedural Knowledge: Task sequences, goals, subgoals, dependencies, constraints.

Task Analysis Questions: Who is going to use the system? What tasks do they perform? What tasks are desired? How are tasks learned? Where are tasks performed? What is relationship between people & data? What other tools do people have? How do people communicate with each other? How often are tasks performed? What are time constraints? What happens when things go wrong?

Selecting and Describing Tasks; **Task Selection Criteria**: Real tasks: From actual user research. Reasonable

coverage: Represent user experience breadth. Difficulty mixture.
Task Description Format: Story style. Include elements. Validation. Example structure.
Hierarchical Task Analysis: Purpose: Break tasks into step-by-step hierarchies. Use: Supports design sketches and storyboards. Structure: Main task → subtasks → detailed steps.
Design Process
Why Sketch?; Key Benefits: Fastest instance of design iteration, Allows for rapid iteration, Enables parallel prototyping, Boosts creativity.
Core Purpose: Sketching is a process that enables thinking through ideas and conveying design concepts early, Sketches are byproducts of the sketching process, Creates feedback loop: mind $<->$ sketch
Important Properties of Sketches: Quick, Timely, Inexpensive, Disposable, Plentiful, Clear Vocabulary, Distinct Gesture, Minimal Detail, Appropriate Degree of Refinement, Suggest and Explore Rather than Confirm, Ambiguity.
Types of Rapid Prototyping: Paper Prototyping, physical Prototyping.
Ethical - Unintended Consequences Framework;
Four Categories of Consequences: Intended by designers + Desired by users/society, Unintended by designers + Desired by users/society, Intended by designers + Undesired by users/society, Unintended by designers + Undesired by users/society.
Robert Merton's 5 Sources of Unintended Consequences: Ignorance - Not knowing enough about the system Human Error - Mistakes in implementation Imperious immediacy of interest - Strong desire for intended outcome leads to ignoring unintended ones Basic values - Your values make you emphasize certain outcomes over others Self-defeating prediction - The prediction itself changes behavior.
Ethical Questions to ask yourself: Do negatives outweigh positives? Are there differential impacts on different populations? Can negative consequences be mitigated vs. inherent to technology? Should we build this at all vs. build it better?
Designing for Diversity and Accessibility; **Core Principles**: WEIRD problem: Designs made for Western, Educated, Industrialized, Rich, Democratic people. Power reconfiguration: Design increases access for some while creating barriers for others. Cultural differences: Website preferences, audio preferences vary significantly across cultures.
Design Approaches: Universal Design vs. Assistive Technology: Universal Design: Built-in accessibility for everyone, Assistive Technology: Separate solutions for specific needs.
Ability-Based Design: Focus on user abilities rather than disabilities, Burden of adaptation falls on the system, not the user, Considers situational limitations.
Design Justice: Goal: Equitably distribute benefits and burdens between groups, Explicitly analyzes oppression across race, gender, disability. Recognizes there is no one-size-fits-all design.
Technical Issues: Algorithmic bias: Sensors and algorithms that discriminate based on skin tone or gender. Audio preferences: Visually impaired users prefer different listening rates than sighted users. Cultural preferences: Website design preferences vary dramatically by culture.
Solutions and Alternatives: **Better Design Strategies**: Adaptive/malleable software: Adjusts to match user abilities, Context-specific designs: Customized for particular situations, Inclusive design teams: Diverse perspectives in development, Diverse user research: Testing with varied populations.
"Better Building" vs. Not Building: Better building: Improving problematic tech, Not building: Refusing to create harmful tech, Worker advocacy: Amazon employees demanding cancellation of facial recognition contracts with law enforcement.
Regulation Trade-offs: Positive: Can drive innovation in alternatives, Provides consumer protection. Negative: Can entrench big players who can afford compliance costs, Can slow innovation by adding hurdles.
What are Storyboards?: Definition: Series of drawn panels showing how a user progresses through a task Visual storytelling with rough sketches/cartoons.
Why Use Storyboards?: Explain experiences to people who haven't experienced them, Put a human face on analytic data, Spark new design concepts and encourage innovation, Share ideas, Give insight into people who are not like us, Persuade others.
Stories Provide Context: Characters, Setting, Sequence, Satisfaction.
Storyboards help think more deeply about specific details about environments where system is used, physical constraints, relationship among multiple people, feelings and concerns of people.
Benefits of Storyboards: Illustrate time, Allow exploration, Effective communication, Can explore more potential approaches.
5 Key Considerations: 1. Level of Detail: too much detail can lose universality, keep simple. 2. Inclusion of Text: Text is often necessary—but keep it short! add captions under frames for clarity. 3. Inclusion of People and Emotions: Show emotional responses (happy, frustrated, confused faces). 4. Number of Frames: Guideline: 4-6 frames per scenario. 5. Portrayal of Time: Use time indicators only when duration matters to the story.
Visual Techniques: Selective use of color - highlight important elements, Blur unnecessary detail in photos.
When to Use Photos: Only if really necessary, Blur out unnecessary detail.
Paper Prototyping; **Prototyping Fidelity**: High Fidelity: Prototypes look and feel more like the final product, More detailed and polished.
Low Fidelity: Designer sketches with many details missing, Paper prototypes fall into this category.
Paper Prototypes Definition: Different sketches of screen appearance on paper, Interactive prototypes using physical paper pieces, Different pieces show different views, dialog boxes and menus, User interacts by writing and pointing, A person simulates the computer's operation.
Why Paper Prototype?: Can make it fairly quickly, Easier to change between/during user tests, No code investment, Focuses attention on big picture, Designer doesn't waste time on details, Users make more creative suggestions, Only kindergarten-level crafting skills required.
Interface Evaluation Methods: Three Main Types: UI Inspection - Expert review methods, Usability Testing - Observing users with tasks, Formal User Testing - Scientific experiments with statistical analysis.
What is UI Inspection? Systematic approaches for experts to evaluate design, Rational (not empirical) evaluation method, Done before user testing to catch obvious problems, Saves study participants for less obvious issues.
Types of UI Inspection: **1. Heuristic Evaluation**: Developed by Jakob Nielsen, 3-5 evaluators examine interface independently, Check compliance with design principles, Focus on product as a whole.
2. Cognitive Walkthrough: Expert identifies user goal and tasks, Focus on specific tasks (not whole product), Expert works through tasks asking 4 questions from user perspective.
Cognitive Walkthrough Questions (at each step): Will user try to achieve the right outcome? Will user notice the correct action is available? Will user associate correct action with expected outcome? Will user see progress toward intended outcome?
What is Usability Testing?; Give participants tasks and observe them perform with prototype.
Why Usability Test? Collects empirical data from non-experts.
What is Formal User Testing? Scientific experiment approach to prove hypothesis, Uniform environment and tasks across participants, Collect quantitative data, May include baseline/alternative prototypes for comparison. Why Formal User Test? Scientific rigor with empirical data Can prove one design is better than others Slower than usability tests.
Examples of Formal Testing Methods: Interviews/Focus Groups - Qualitative & Moderated, Surveys - Qualitative/Quantitative & Unmoderated, Clickstream Analysis - Quantitative & Unmoderated, Eye-tracking Studies - Quantitative & Unmoderated, Lab Studies - Qualitative/Quantitative & Moderated, Remote Testing - Various formats, A/B Experiments - Quantitative & Unmoderated.
How A/B Testing Works: Split users 50/50 between control and treatment. Control: existing system. Treatment: existing system with new feature.
A/B Testing Example: Amazon shopping cart recommendations.
Limitations of A/B Tests: Drives hill-climbing, not overall design innovation, Only marginal improvements, No generalizable knowledge gained, Novel designs can't compete fairly, Difficult to scale for many comparisons, Over-reliance on quantitative metrics can mislead.

What is Heuristic Evaluation? Nielsen's 10 Heuristics;
1. Visibility of System Status: Keep users informed about what's happening, Provide appropriate feedback within reasonable time.
2. Match Between System and Real World: Speak users' language, Use familiar words, phrases, concepts, Follow real-world conventions, Natural and logical order of information.
3. User Control and Freedom: Provide "emergency exits" from unwanted states, Support undo and redo functionality, Don't force users through extended dialogues, Users need to escape any situation or state.
4. Consistency and Standards: Same words/actions = same meaning throughout, Follow platform conventions, Internal consistency: within same product, External consistency: with other products in same class.
5. Error Prevention: Better than good error messages = prevent errors entirely, Eliminate error-prone conditions when possible, Confirmation dialogs before committing to actions, Check for problems and present options before user commits.
6. Recognition Rather Than Recall: Make objects, actions, options visible, Don't make users remember info from one part to another, Instructions should be visible or easily retrievable, Hidden affordances: need to remember where to do something, False affordances: need to remember right way to do something.
7. Flexibility and Efficiency of Use: Cater to novice AND expert users, Provide accelerators for experts, Allow multiple ways to do things, Avoid repetitive manual actions, Let users automate frequent actions.
8. Aesthetic and Minimalist Design: Remove irrelevant/rarely needed information, Every extra unit competes with relevant information, Not just about ugliness - about clutter, overload, visual noise, Avoid distracting animations.
9. Help Users Recognize, Diagnose, and Recover from Errors: Plain language, Precisely indicate the problem, Constructively suggest solutions, Error messages should be helpful, not cryptic.
10. Help and Documentation: Better if system works without documentation, When needed, make it: Easy to search, Focused on user's task, List concrete steps, Not too large.
Visual Design
C.R.A.P. Design Principles: Main goal: Simplicity - reduce inessential features for easier learning and faster use.
Contrast: Definition: Imply difference in meaning by difference in style, Elements meant to be same should NOT look "slightly different", Elements that are different should look VERY different.
Repetition: Definition: Repeat visual elements to create unity and cohesiveness. Good candidates: color palette, typefaces, graphic styles. Creates visual connection between elements.
Alignment: Definition: Align elements to create visual connections and unity. Nothing should be placed arbitrarily. Use grids for clean, organized look. Columns make horizontal scanning easier.
Proximity: Definition: Group related elements, put unrelated things far apart. Create visual units rather than separate items. Users shouldn't have to work hard to figure out relationships. Be aware of unintentional groupings.
Gestalt Principles: Core idea: Human brain simplifies and organizes designs into whole patterns Key Principles. Proximity: Objects close together are perceived as grouped. Similarity: Similar objects form groups. Closure: Incomplete objects perceived as whole. Symmetry/Order: Objects perceived as symmetrical around center point. Continuity: Objects perceived as grouped when they align.
Balance and Symmetry: Small scale: exact mirror-image equivalence. Large scale: balance of "stuff" on each axis. "Stuff" measured by mass (quantity of pixels) and extent (area covered).
Negative Space: Use for grouping instead of lines. Use margins to guide eye movement. Don't crowd controls together. Crowding creates spatial tension and inhibits scanning.
Color Guidelines; Basic Rules: Avoid saturated colors. Use few colors.
Color Vision Deficiency: Red-green deficiency: affects 8% of population. Blue-yellow deficiency: much rarer. Solution: Use redundant coding, don't depend only on color. Avoid red-green distinctions.
Typography; Key Terms: Typography: art and science of displaying text. Font: shapes of letters and characters. Spacing: white space around letters, words, lines, paragraphs. Kerning: adjusting character spacing for specific character pairs.
Text Spacing Rules: Always leave margins around body text, Use generous leading, Keep paragraphs narrow, Don't overcrowd body text.
Font Selection: Limit: 2-3 typefaces maximum, Don't use two faces from same category, Use size, weight, style, hue for contrast, Maximum 4-5 font varieties total.
What is Disability?: **1980 Definition**: Disability as a deficit - "restriction or lack of ability to perform an activity in the manner considered normal".
Today's Definition: Disability as context dependent - "complex phenomenon reflecting interaction between person's body and society features".
Principles of Inclusive Design
Recognizing Exclusion: Investigate mismatched interactions and design solutions.
Learn From Diversity: Understand how people adapt to the world around them.
Solve for One, Extend to Many: Designing for permanent disabilities benefits larger populations.
Accessibility: Focuses on specific disabilities and compliance with standards, end outcome focused - ensuring products meet accessibility standards.
Web Content Accessibility Guidelines (WCAG): Internationally accepted standards for web accessibility.
Four Main Principles: Perceivable: Information must be presentable in ways users can perceive. Operable: User interface components must be operable. Understandable: Information and operation must be understandable. Robust: Content must be robust enough for interpretation by assistive technologies.
Key WCAG Guidelines: Contrast, Use of Color, Input Assistance, Section Headings, Non-Text Content, Focus Order, Focus Visible.
Animation Principles: Squash and Stretch: Object deformation shows impact/flexibility. Anticipation: Small movement before main action. Slow In, Slow Out: Natural acceleration/deceleration curves.
UI Design Patterns: General repeatable solutions to commonly-occurring usability problems.
Accessibility Testing Methods: Automated Tools: Can catch basic issues but not everything. Keyboard Navigation Testing: Try navigating with only keyboard. Screen Reader Testing: Test with actual assistive technology. Color/Contrast Testing: Use grayscale filters, contrast checkers. User Testing: Include users with disabilities in testing process.
Red Flags to Check: Images without alt text, Low color contrast, Buttons/links without clear focus states, Forms without proper labels, Content that can't be navigated by keyboard, Relying only on color to convey information.